*V2.0-Beta2*

Modification Records

* 参考Management Entity的概念，建立MD，MA的模型，不再使用简单的MDLevel，MDName，MDID/MAName，MAID来描述二者的功能和属性
* 不再使用一个结构体MP来同时描述MEP与MIP，对二者分别建立各自的结构
* 设备上应该有PORT及可通过该PORT的VLAN的概念，现模拟化低将其存储于FiteringDatabase中；对于不同PORT最先加入FiteringDatabase的MAC被认为是本设备上对应于该PORT的MAC地址，用于MIP的创建
* MIP不再人工手动配置，根据协议描述，设备上管理实体MD，MA，MEP及其之间相互关系的改变应触发MIP的创建。
* MD/MA/MEP/MIP层次关系:

CFM

MD List

MA List

MEP List

MIP List

MD1

MD2

MD3

...

MA1

MA2

MA3

...

MEP1

MEP2

MEP3

...

MIP1

MIP2

MIP3

...

* 虽然在存储上MD，MA，MEP，MIP是同层次的，但是由于MA内有MD 的引用，MEP/MIP内有MA的引用，故在逻辑上是分层次的。对高层次的对象进行删除操作将同时删除下层的对象
* User Space仅支持获取MD的List，对MA/MEP/MIP的获取只能依靠各自的meid
* MD/MA/MEP之间有一定的校验规则，为此增加*ConfigurationErrorList*，在MIP的同时检查配置，将配置错误加入列表；通过调用API可获得特定PORT/VID上的错误
* *ChassisManagement*对象仍存储于Cfm下的SenderIDTLV结构中，调用API可分别获取特定项的值也可一次获取全部值
* *isMA\_no\_UpMEP\_on\_Port,isMA\_no\_MEP\_on\_Port,isMA\_no\_DownMEP,isMA\_no\_UpMEP,do\_MAs\_have\_common\_vid,UpdateErrorList*分步骤更新ConfigurationErrorList
* *CheckCondition\_1, CheckCondition\_2, CheckCondition\_3, CheckCondition\_4*为根据协议实现的MIP创建条件检查函数，这里根据情况为实现第一个函数；*cfm\_createMIPs, UpdateMIPS*函数将从*FiteringDatabase*中取得PORT/VID对依次检查
* 一个API调用可能引起多次的MIP List的更新操作，删除的MIP要释放内存
* 创建MIP与否取决于特定PORT/VID对应的MA的*MHFCreation*值，如果意外未能取得，则从*DefaultMDLevel*中获取；默认的MA取值是不创建MIP
* SenderIDTLV的编码与否应取决于MA的*SenderIDPermission*值，但是LBP/LTP内仍先要取决于lbpm/ltpm类似定义的值，默认情况下都是不添加SenderIDTLV(None)；

CCP中取决于MA的*SenderIDPermission*值，默认情况下不添加SenderIDTLV(None)

* 创建MEP时要加入*MPStatusList*，删除MEP时要除去对应项
* Port改用unsigned short int(uint16)表示，Flow仍用unsigned int(uint32)表示
* 对MD/MA/MEP等的*Set*操作，根据参数值(event)对结构中不同属性值操作
* 根据实际需要，在*CreateMEP*操作中所需的Layer 2 entity pointer简化为包含port值和MAC地址的结构
* FiteringDatabase结构经过重新定义，关系到PORT，VLAN与MAC
* MEP中的MEPStatus定义的计数器/存贮单元是协议机构体(lbpm/ltpm/ccpm)的子集，暂并未真正使用
* Kernel与User通过在帧前加port与flow进行交流，由于port从uint32改为uint16，前端字节数也从8个字节改为6个字节

|  |
| --- |
| 本程序经过修改及测试，外部框架得到了更新，保留或改进了之前版本的功能，并增加了更多的API。下面问题需说明：   1. 我们一直使用FilteringDatabase来模拟设备上端口和MAC地址情况，这里需要将测试网络上所有MAC都加进去，并且，本设备上MAC与PORT对要最先加进去，加进去后程序启动后将不能改变。这种设置使在PORT p上自动创建的MIP可获取对应的MAC地址； 2. MIP现根据设备上的PORT与VLAN信息（储存在FilteringDatabase中）进行自动创建。但是预设情况下某MA下是不创建MIP的，通过Api：cfmSupervisory\_SetMA修改MHF\_Creation\_Default属性值可创建MIP； 3. 创建MIP后自动进行Configuration-Error的检测 4. MD，MA，MEP的创建，修改与删除可在程序运行前配置（cfm\_core.c），也可在运行后通过User Space 调用Api实现，任何一种调用都将激发MIP的刷新； 5. 测试前，需对cfm\_core\_init（cfm\_core.c）中进行相关设置，包括各种MD，MA，MEP和创建MEP的port与MAC信息，另外FilteringDatabase\_init（FilteringDatabase.c）中对网络上各设备端口相对与本设备的位置及MAC也需手动设置，example.c中有对各个API进行测试的程序段，以#if 0~#endif包围，里面参数值可供修改，example.c程序支持1个port与2个port情况两种模式，由宏TWONIC控制； 6. 现实现扔将网络中抓到的所有数据帧全部写进内核，测试时可根据需要修改write函数附近代码(example.c)只将CFM帧写入，与md5相关的操作用于转发数据帧中避免抓回自己发出的帧； 7. example.c需要系统中网络为eth0和eth1，请根据具体情况修改。 |